Web 2.0, allez plus loin avec AJAX et XMLHttpRequest

par siddh

Date de publication : 07/01/2006

Dernière mise à jour : 15/11/2006

Si il y a un terme à la mode en ce moment c'est bien celui d'AJAX, on le retrouve un peu a toutes les sauces. Qu'est ce donc que cet AJAX ? Nous verrons à quoi correspondent ces lettres mais surtout les concepts qui se cachent derrière et les différentes utilisations que l'on peut en faire.

I - Introduction

Qu'est ce qu'AJAX ?

Vous trouverez certainement plusieurs définitions différentes à cet acronyme.

En voici une : **Asynchronous JavaScript And XML.**

AJAX n'est donc pas une technologie mais plutôt un concept qui permet donc de faire des appels asynchrones au serveur depuis le client.

Lors de ces appels, le serveur retournera du XML qui sera "récupéré" par javascript et traité.

Nous verrons que nous pouvons tout aussi bien faire transiter du texte et faire des appels synchrone si l'on veut.

Avant toute chose, il serait bon de faire un point sur le processus classique de consultation d'un site ou d'une application web :

1 Vous saisissez une adresse dans votre navigateur.
2 Cette "requête" finie par arriver sur le serveur web qui héberge la page en question.
3 Le serveur vous retourne du texte au format HTML ou XHTML et éventuellement des images, feuilles de style, fichiers JavaScript, applets java ....
4 Votre navigateur les interprète et vous affiche la page.
5 Vous êtes déconnecté du serveur web.

Donc, quand vous cliquez sur un lien, vous recommencez ce processus en entier avec une nouvelle page.

Dans le cas ou un formulaire se trouve sur la page, vous envoyez les données sur le serveur qui vous répondra après traitement de ces données.

L'utilisation d'AJAX va chambouler un peu cette organisation car a tout moment vous pouvez aller chercher des informations sur le serveur pour :

- Ajouter des éléments a la page
- Modifier le contenu d'un "bout de la page"
- Insérer des données dans une base.

Les applications sont donc très nombreuses :

- Champs qui s'auto-complètent (comme google suggest).
- Listes déroulantes liées.
- Contrôle de formulaire.
- Popups accessibles.
- ...

Mais comment faire pour l'utiliser ?

II - L'objet XmlHttpRequest

AJAX se base sur l'utilisation d'un composant embarqué dans presque tous les navigateurs récents.

Par contre, vous vous doutez bien que le comportement va varier en fonction de ces derniers.

Pour pouvoir utiliser AJAX, il nous faut donc créer en javascript un objet que l'on nomme **XmlHttpRequest** ou **xhr** pour les intimes, comme son nom l'indique, il va nous permettre de faire des requêtes http pour échanger du XML.

Création de l'objet XmlHttpRequest

```
var xhr = null;

if(window.XMLHttpRequest) // Firefox et autres
   xhr = new XMLHttpRequest();
else if(window.ActiveXObject){ // Internet Explorer
   try {
        xhr = new ActiveXObject("Msxml2.XMLHTTP");
    } catch (e) {
        xhr = new ActiveXObject("Microsoft.XMLHTTP");
    }
}
else { // XMLHttpRequest non supporté par le navigateur
   alert("Votre navigateur ne supporte pas les objets XMLHTTPRequest...");
   xhr = false;
}
```

Comme vous pouvez le constater, Internet Explorer se démarque des autres.

A son crédit on peut quand même préciser que rien n'est standardisé à ce niveau là et que même si c'est un ActiveX, il ne fait pas apparaître de message de sécurité.

III - Utilisation

Nous avons donc notre xhr qui est prêt a l'emploi.

Pour l'utiliser, nous disposons de différentes propriétés et méthodes :

Propriété ou méthode	Description
open("méthode","url",flag)	Ouvre la connexion avec le serveur. **méthode** -> "GET" ou "POST" **url** -> l'url à laquelle on va envoyer notre requête. Si la méthode est GET, on met les paramètres dans l'url. **flag** -> true si l'on veut un dialogue asynchrone, sinon, false
setRequestHeader("nom","valeur")	Assigne une valeur à un header HTTP qui sera envoyé lors de la requête. Par exemple, pour un POST : nom -> "Content-Type" valeur -> "application/x-www-form-urlencoded"
send("params")	Envoi la requête au serveur. Si la méthode est GET, on met **null** en paramètre. Si la méthode est POST, on met les paramètres a envoyer, sous la forme : "nomparam1=valeurparam1&nomparam2=valeurparam2".
abort()	Abandonne la requête.
onreadystatechange	Ici, on va lui affecter une fonction a nous qui sera exécutée à chaque "changement d'état" de notre objet.
readyState	C'est cette propriété qu'on va tester dans le onreadystatechange. Elle représente l'état de l'objet et peut prendre plusieurs valeurs : 0 -> Non initialisé. 1 -> Ouverture (open() vient de s'exécuter). 2 -> Envoyé (send() vient de s'exécuter). 3 -> En cours (des données sont en train d'arriver).

Propriété ou méthode	Description
	4 -> Prêt (toutes les données sont chargées).
status	Le code de la réponse du serveur. 200 -> OK. 404 -> Page non trouvée. ... Liste complète
statusText	Le message associé à status.
responseText	La réponse retournée par le serveur, au format texte.
responseXML	La réponse retournée par le serveur, au format dom XML.

Maintenant que nous avons vu tout ça, rien de tel qu'un petit exemple :

index.html

```
<html>
        <head>
                <title>Tutoriel Ajax (XHTML + JavaScript + XML)</title>
                <script type='text/JavaScript'>

                        function getXhr(){
                                var xhr = null;
                                if(window.XMLHttpRequest) // Firefox et autres
                                   xhr = new XMLHttpRequest();
                                else if(window.ActiveXObject){ // Internet Explorer
                                   try {
                                        xhr = new ActiveXObject("Msxml2.XMLHTTP");
                                    } catch (e) {
                                        xhr = new ActiveXObject("Microsoft.XMLHTTP");
                                    }
                                }
                                else { // XMLHttpRequest non supporté par le navigateur
                                   alert("Votre navigateur ne supporte pas les objets
XMLHTTPRequest...");
                                   xhr = false;
                                }
                                return xhr
                        }

                        /**
                        * Méthode qui sera appelée sur le click du bouton
                        */
                        function go(){
                                var xhr = getXhr()
                                // On défini ce qu'on va faire quand on aura la réponse
                                xhr.onreadystatechange = function(){
                                        // On ne fait quelque chose que si on a tout reçu et que le
serveur est ok
                                        if(xhr.readyState == 4 && xhr.status == 200){
                                                alert(xhr.responseText);
                                        }
                                }
                                xhr.open("GET","ajax.php",true);
                                xhr.send(null);
                        }
```

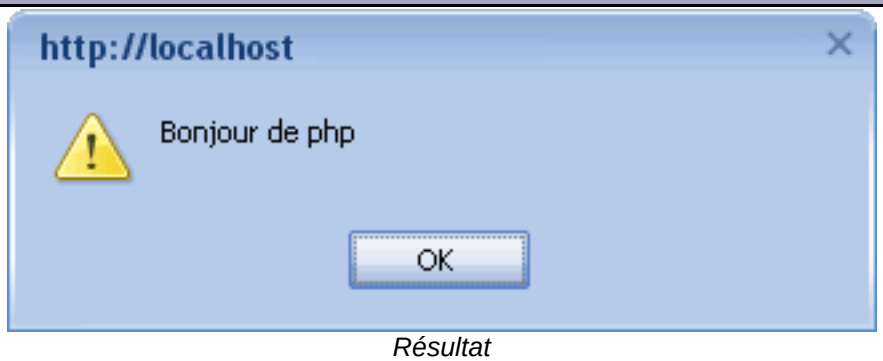
index.html

```
                </script>
        </head>
        <body>
                <input type='button' value='Dis quelque chose !' onclick='go()' />
        </body>
</html>
```

ajax.php

```
<?php
        echo "Bonjour de php";
?>
```

Le résultat



Résultat

Après avoir fait ce test, vous pouvez aller changer la réponse de php et recliquer sur le bouton sans recharger la page.

Vous obtiendrez la nouvelle réponse !.

La construction de l'objet est faîte dans une fonction car sinon dans Internet Explorer, on ne pourra pas "enchaîner" plusieurs exécutions.

IV - Aller plus loin

IV-A - Listes liées

Imaginons que l'on ai dans notre base de donnée une table Auteur et une table Livre.

Le but du jeu est d'avoir une liste contenant les auteurs qui permette de filtrer celle des livres.

Faisons simple :

Auteur contiendra les champs id et nom.

Livre aura les champs id, titre et auteurId.

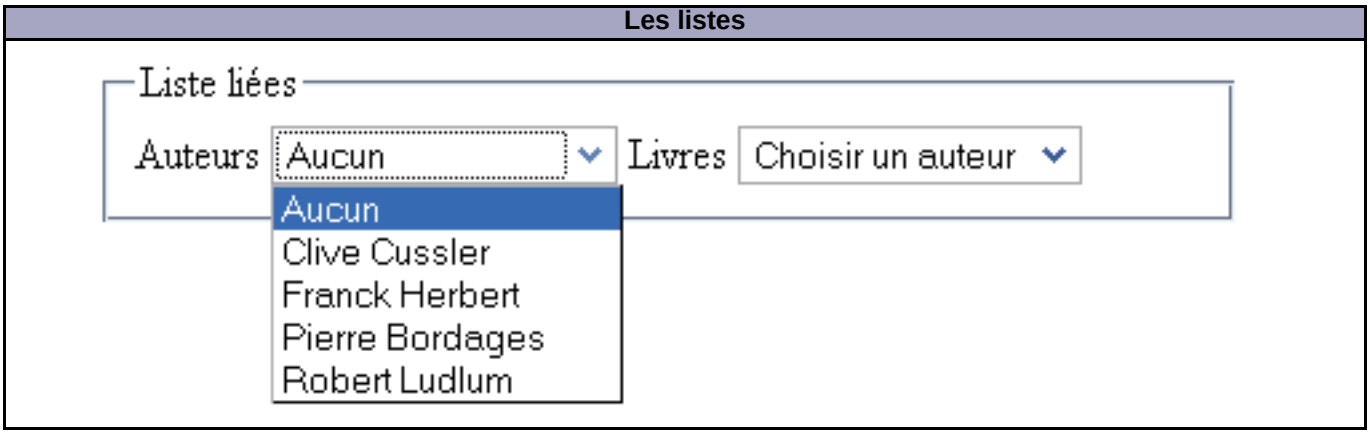
```
CREATE TABLE `auteur` (
  `id` tinyint(4) NOT NULL auto_increment,
  `nom` varchar(50) NOT NULL,
  PRIMARY KEY  (`id`)
);

insert into `auteur` values
(1,'Clive Cussler'),
(2,'Harlan Coben'),
(3,'Franck Herbert'),
(4,'Pierre Bordages');

CREATE TABLE `livre` (
  `id` tinyint(4) NOT NULL auto_increment,
  `titre` varchar(50) NOT NULL,
  `idAuteur` tinyint(4) default NULL,
  PRIMARY KEY  (`id`)
) ;

insert into `livre` values
(1,'Odyssee',1),
(2,'Sahara',1),
(3,'Dragon',1),
(4,'Une chance de trop',2),
(5,'Ne le dis a personne',2),
(6,'Disparu à jamais',2),
(7,'Dune',3),
(8,'La barriere de santaroga',3),
(9,'Les guerriers du silence',4),
(10,'La citadelle hyponeros',4),
(11,'Terra mater',4);
```

Les listes



Les listes

Liste auteurs

Liste livres

auteurs.php

```
<html>
        <head>
                <title>Tutoriel Ajax (XHTML + JavaScript + XML)</title>
                <script type='text/javascript'>

                        function getXhr(){
                                var xhr = null;
                                if(window.XMLHttpRequest) // Firefox et autres
                                   xhr = new XMLHttpRequest();
                                else if(window.ActiveXObject){ // Internet Explorer
                                   try {
                                        xhr = new ActiveXObject("Msxml2.XMLHTTP");
                                    } catch (e) {
                                        xhr = new ActiveXObject("Microsoft.XMLHTTP");
                                    }
                                }
                                else { // XMLHttpRequest non supporté par le navigateur
                                   alert("Votre navigateur ne supporte pas les objets
XMLHTTPRequest...");
                                   xhr = false;
                                }
                                return xhr;
                        }

                        /**
                        * Méthode qui sera appelée sur le click du bouton
                        */
                        function go(){
                                var xhr = getXhr();
                                // On défini ce qu'on va faire quand on aura la réponse
                                xhr.onreadystatechange = function(){
                                        // On ne fait quelque chose que si on a tout reçu et que le
serveur est ok
                                        if(xhr.readyState == 4 && xhr.status == 200){
                                                leselect = xhr.responseText;
                                                // On se sert de innerHTML pour rajouter les options
a la liste
                                                document.getElementById('livre').innerHTML =
leselect;
                                        }
                                }

                                // Ici on va voir comment faire du post
                                xhr.open("POST","ajaxLivre.php",true);
                                // ne pas oublier ça pour le post
                                xhr.setRequestHeader('Content-Type','application/x-www-form-urlencoded');
                                // ne pas oublier de poster les arguments
                                // ici, l'id de l'auteur
                                sel = document.getElementById('auteur');
                                idauteur = sel.options[sel.selectedIndex].value;
                                xhr.send("idAuteur="+idauteur);
                        }
                </script>
        </head>
        <body>
                <form>
```

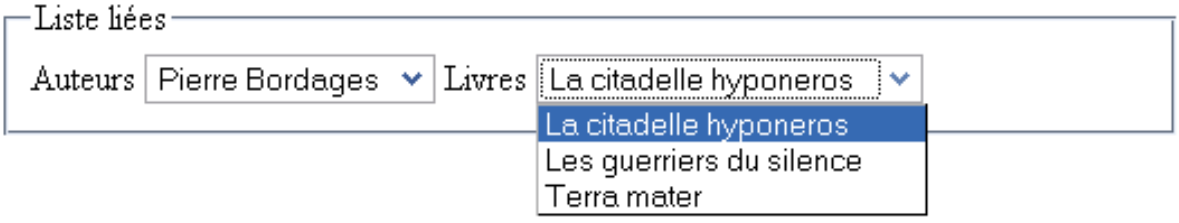
auteurs.php

```
                        <fieldset style="width: 500px">
                                <legend>Liste liées</legend>
                                <label>Auteurs</label>
                                <select name='auteur' id='auteur' onchange='go()'>
                                        <option value='-1'>Aucun</option>
                                        <?
                                                mysql_connect("localhost","root","root");
                                                mysql_select_db("test");
                                                $res = mysql_query("SELECT * FROM auteur ORDER BY
nom");
                                                while($row = mysql_fetch_assoc($res)){
                                                        echo "<option
value='".$row["id"]."'>".$row["nom"]."</option>";
                                                }
                                        ?>
                                </select>
                                <label>Livres</label>
                                <div id='livre' style='display:inline'>
                                <select name='livre'>
                                        <option value='-1'>Choisir un auteur</option>
                                </select>
                                </div>
                        </fieldset>
                </form>
        </body>
</html>
```

ajaxLivre.php

```
<?php
        echo "<select name='livre'>";
        if(isset($_POST["idAuteur"])){
                mysql_connect("localhost","root","root");
                mysql_select_db("test");
                $res = mysql_query("SELECT id,titre FROM livre
                        WHERE idAuteur=".$_POST["idAuteur"]." ORDER BY titre");
                while($row = mysql_fetch_assoc($res)){
                        echo "<option value='".$row["id"]."'>".$row["titre"]."</option>";
                }
        }
        echo "</select>";
?>
```

Le résultat



Résultat des listes

C'est un code minimaliste qui ne contient pas tous les contrôles, notamment au niveau php mais cela permet de se faire une petite idée.

Si on doit mettre le select dans un div et faire un innerHTML dessus, c'est car Internet Explorer ne gère pas les innerHTML dans un select.

IV-B - Utiliser responseXML

Pour l'instant, nous nous sommes contentés d'utiliser la réponse au format texte et ensuite soit de l'afficher tel quel ou de s'en servir avec un innerHTML.

Mais je vous rappel que le X de AJAX veut dire XML.

Jusqu'à présent nous n'avons donc pas fais d'AJAX mais juste de l'utilisation de XmlHttpRequest. Ce qui n'est pas très grave mais qui fera hurler les puristes.

ne vous amusez pas à dire que vous faites de l'AJAX dans ces cas là ;).

Pour utiliser le responseXML, il va falloir que notre serveur nous retourne du XML.

il faut donc que notre XML soit "bien formé" mais aussi que votre navigateur l'interprète comme du XML.

Il nous faudra donc rajouter les headers adéquats.

Malheureusement, les choses à gérer ne s'arrêtent pas là.

Comme le fais si bien remarquer denisC dans la Faq javascript, nous allons devoir "nettoyer" notre XML.

J'ai retenu la solution suivante qui est le fruit de denisC et javatwister :

node cleaner
```
function go(c){
        if(!c.data.replace(/\s/g,''))
                c.parentNode.removeChild(c);
}

function clean(d){
        var bal=d.getElementsByTagName('*');

        for(i=0;i<bal.length;i++){
                a=bal[i].previousSibling;
                if(a && a.nodeType==3)
                        go(a);
                b=bal[i].nextSibling;
                if(b && b.nodeType==3)
                        go(b);
        }
        return d;
}
```

Et c'est le documentElement de notre responseXML qu'il faudra transmettre a notre nettoyeur.

Ensuite, on parcours notre résultat en utilisant le dom.

indexxml.html
```
<html>
        <head>
                <title>Tutoriel Ajax (XHTML + JavaScript + XML)</title>
                <script type='text/JavaScript'>

                        function getXhr(){
                                var xhr = null;
                                if(window.XMLHttpRequest) // Firefox et autres
                                   xhr = new XMLHttpRequest();
                                else if(window.ActiveXObject){ // Internet Explorer
                                   try {
```

indexxml.html

```
                                        xhr = new ActiveXObject("Msxml2.XMLHTTP");
                                    } catch (e) {
                                        xhr = new ActiveXObject("Microsoft.XMLHTTP");
                                    }
                                }
                                else { // XMLHttpRequest non supporté par le navigateur
                                   alert("Votre navigateur ne supporte pas les objets
XMLHTTPRequest...");
                                   xhr = false;
                                }
                                return xhr;
                        }

                        // Node cleaner
                        function go(c){
                                if(!c.data.replace(/\s/g,''))
                                        c.parentNode.removeChild(c);
                        }

                        function clean(d){
                                var bal=d.getElementsByTagName('*');

                                for(i=0;i<bal.length;i++){
                                        a=bal[i].previousSibling;
                                        if(a && a.nodeType==3)
                                                go(a);
                                        b=bal[i].nextSibling;
                                        if(b && b.nodeType==3)
                                                go(b);
                                }
                                return d;
                        }

                        /**
                        * Méthode qui sera appelée sur le click du bouton
                        */
                        function gophp(){
                                var xhr = getXhr();
                                // On défini ce qu'on va faire quand on aura la réponse
                                xhr.onreadystatechange = function(){
                                        // On ne fait quelque chose que si on a tout reçu et que le
serveur est ok
                                        if(xhr.readyState == 4 && xhr.status == 200){
                                                reponse = clean(xhr.responseXML.documentElement);
                                                alert(reponse.getElementsByTagName("message")[0].firstChild.node
                                        }
                                }
                                xhr.open("GET","ajaxxml.php",true);
                                xhr.send(null);

                        }
                </script>
        </head>
        <body>
                <input type='button' value='Dis quelque chose !' onclick='gophp()' />
        </body>
</html>
```

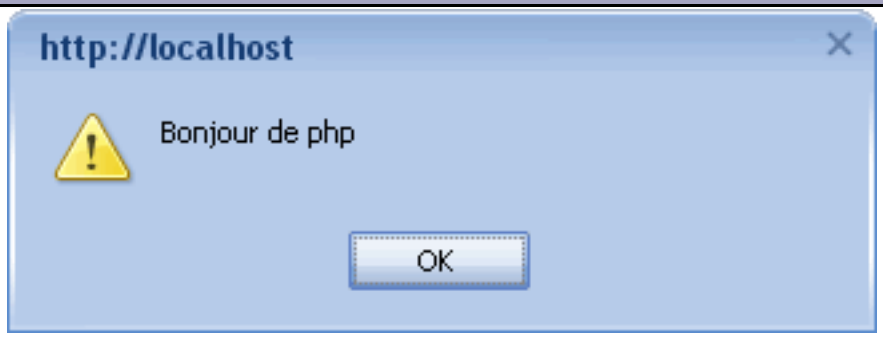
ajaxxml.php

```
<?php
   $buffer  = '<?xml version="1.0"?>';
   $buffer .= "<reponse><message>Bonjour de PHP &amp;amp; d'XML</message></reponse>";

   header('Content-Type: text/xml');
   echo $buffer;
?>
```

Le résultat

Le résultat



Résultat

V - Conclusion

L'utilisation de XmlHttpRequest va vous permettre d'améliorer l'interactivité entre votre page et l'utilisateur. il ne faut pas perdre de vue que son utilisation dépend du fait que javascript doit être activé sur le navigateur. Il faut donc prévoir éventuellement une solution de repli.

En ce qui concerne l'utilisation synchrone ou asynchrone, si vous faites du synchrone, vous risquez de vous retrouver avec une page bloquée qui attendra le retour du serveur. A manipuler avec prudence donc.

Aujourd'hui tout le monde se met à utiliser cette technologie. Faire de l'AJAX pour en faire ne sert à rien, il faut que ça apporte vraiment un plus pour l'utilisation de votre site ou de votre application.

Il existe déjà des frameworks JavaScript intégrant AJAX, c'est le cas notamment de rico et prototype.